What is claimed is:

1. A method for compressing an image, the image including pixels arranged along a plurality of raster lines, the method comprising the steps of:

dividing select pixels in the image into a plurality of slices, the slices including pixels positioned adjacent to one another;

representing each slice in the form of an input slice-value;

comparing each input slice-value to determine if the input slice-value corresponds to a memory slice-value in a memory store; and

generating an output based on the comparing, the generating resulting in a compressed image.

2. The method of claim 1, wherein the memory store is a cache memory.

3. The method of claim 1, wherein the step of generating an output includes the step of outputting at least one of an output slice-value and an output encoded-value.

4. The method of claim 3, wherein the generating step includes:

generating an output slice-value if the input slice-value does not correspond to a memory slice-value in a memory store; and

generating an output encoded-value if the input slice-value does correspond to a memory slice-value in a memory store.

5. The method of claim 1, wherein generating an output based on the comparing includes the output being a least partially a series of repeating values, the output being further processed with run-length encoding.

6. The method of claim 1, wherein each slice is a two-dimensional array of pixels.

7. The method of claim 6, wherein the two-dimensional array of pixels defines a rectangle, the rectangle possessing a width w and a height h.

8. The method of claim 1, wherein the memory store possesses a plurality of addresses including a lowest address, and wherein the step of comparing each input slice-value to determine if the input slice-value corresponds to a memory slice-value in a memory store includes determining if the corresponding memory slice-value is disposed at the lowest address in the memory store.

9. The method of claim 8, wherein if the corresponding memory slice-value is disposed at the lowest address in the memory store, then the comparing step further includes

determining if a predetermined address above the lowest address in the memory store corresponds to the lowest address.

10. The method of claim 1, wherein the memory store possesses a plurality of addresses, and wherein the step of comparing each input slice-value to determine if the input slice-value corresponds to a memory slice-value in a memory store includes:

initially comparing each input slice value to a base cache-address, the location of the base cache-address determined in a previous comparison; and

comparing, if no match is found at the base cache-address, the input slice-value to the lowest address in the memory and moving upwards until a match is found.

11. The method of claim 10, wherein if no match is found in the memory store, then the step of generating an output including generating an output slice-value,

the method further including the step of storing the input slice value in the memory store, the storing including:

storing the input slice-value in the lowest address in the memory store while shifting the content of the memory store upwards; and

assigning the lowest address in the memory store as the base cache-address, the base cache-address being the first address that the comparing process will compare in processing a next input slice-value.

12. The method of claim 10, wherein if a match is found at the lowest address in the memory store, then the step of generating an output including generating an output encoded-value identifying the lowest address location, and the comparing step further includes determining if a predetermined address above the lowest address in the memory store corresponds to the lowest address, and if the predetermined address above the lowest address does not match the lowest address, then

the method further including the step of storing the input slice value in the memory store, the storing including:

storing the input slice-value in the lowest address in the memory store while shifting the content of the memory store upwards; and

assigning the lowest address in the memory store as the base cache-address, the base cache-address being the first address that the comparing process will compare in processing a next input slice-value.

13. The method of claim 10, wherein if a match is found at the lowest address in the memory store, then the step of generating an output including generating an output

encoded-value identifying the lowest address location, and the comparing step further includes determining if a predetermined address above the lowest address in the memory store corresponds to the lowest address, and if the predetermined address above the lowest address does match the lowest address, then

the method further including the step of assigning the lowest address in the memory store as the base cache-address, the base cache-address being the first address that the comparing process will compare in processing a next input slice-value.

14. The method of claim 10, wherein if a match is found but not at the lowest address in the memory store, then the step of generating an output including generating an output encoded-value identifying the address location of the match, and

the method further including the step of moving the matching slice-value in the memory store, the moving including:

moving the matching slice-value in the memory store to the lowest address in the memory store; and

moving all the contents of all addresses in the memory which are lower than the matching address up one address; and

assigning the matching address in the memory store as the base cache-address, the base cache-address being the first address that the comparing process will compare in processing a next input slice-value.

15. The method of claim 1, wherein the memory store possesses a plurality of addresses and the method for compressing an image further includes subsequently decompressing the compressed image, the decompressing including:

reading an item from the compressed image, the item being one of a slice-value or an encoded-value which represents an address location in the memory store, and if the item is an encoded-value; then

using the encoded-value to read a slice-value from the memory store.

16. The method of claim 15, wherein if the item is a slice-value, then the step of decompressing further including;

reassembling a reconstructed image slice based on the slice-value;

outputting the reconstructed image slice; and

storing the slice-value read from the compressed image in the lowest address in the memory store while shifting the content of the memory store upwards.

17. The method of claim 15, wherein if the item is an encoded-value and the encoded-value corresponds to the lowest address in the memory store, then the step of decompressing further including;

reassembling a reconstructed image slice based on the slice-value read from the memory store based on the encoded-value;

outputting the reconstructed image slice; and

determining if a predetermined address above the lowest address in the memory store corresponds to the lowest address, and if the predetermined address above the lowest address does match the lowest address, then maintaining the content of the memory store.

18. The method of claim 15, wherein if the item is an encoded-value and the encoded-value corresponds to the lowest address in the memory store, then the step of decompressing further including;

reassembling a reconstructed image slice based on the slice-value read from the memory store based on the encoded-value;

outputting the reconstructed image slice; and

determining if a predetermined address above the lowest address in the memory store corresponds to the lowest address, and if the predetermined address above the lowest address does not match the lowest address, then storing the slice-value read from the memory store in the lowest address in the memory store while shifting the content of the memory store upwards.

19. The method of claim 15, wherein if the item is an encoded-value and the encoded value does not correspond to the lowest address in the memory store, then the step of decompressing further including;

reassembling a reconstructed image slice based on the slice-value read from the memory store based on the encoded-value;

outputting the reconstructed image slice; and

storing the slice-value read from the memory in the lowest address in the memory store while shifting the content of the memory store upwards.

20. The method of claim 15, wherein the memory store is initialized similarly before compressing and decompressing an image, and wherein information regarding the content of the memory store in the compression process is not needed in the decompression process.

21. An apparatus for compressing an image, the image including pixels arranged along a plurality of raster lines, the apparatus comprising:

means for dividing select pixels in the image into a plurality of slices, the slices including pixels positioned adjacent to one another;

means for representing each slice in the form of an input slice-value;

means for comparing each input slice-value to determine if the input slice-value corresponds to a memory slice-value in a memory store; and

means for generating an output based on the determination of whether the input slice-value corresponds to a memory slice-value in a memory store, the output forming data of a compressed image.

22. The apparatus according to claim 21, wherein the output is in the form of at least a portion of repeating values.

23. The apparatus according to claim 21, wherein the output is further processed using run-length encoding.

24. The apparatus according to claim 21, wherein the memory store is a cache memory.

25. The apparatus according to claim 21, wherein the means for generating an output outputs at least one of an output slice-value and an output encoded-value; and wherein:

the means for generating an output generates an output slice-value if the input slice-value does not correspond to a memory slice-value in a memory store; and

the means for generating an output generates an output encoded-value if the input slice-value does correspond to a memory slice-value in a memory store.